(1)

ADST-94-W003283

ADST SIMWORLD Data Base

BDS-D Architecture and Standards

Loral Systems Company
12151-A Research Parkway
Orlando, Florida 32826

April 18, 1994

Contract No. N61339-91-D-001
BDS-D Architecture and Standards Phase 2, D.O. 0035
CDRL A006

Prepared for:

U.S. Army Simulation, Training and Instrumentation Command (STRICOM)
12350 Research Parkway
Orlando, FL 32826-3224

LORAL

REPORT DOCUMENTATION PAGE

Form approved OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	**2. REPORT DATE** 4/18/94	**3. REPORT TYPE AND DATES COVERED**

4. TITLE AND SUBTITLE
Advanced Distributed Simulation Technology
SIMWORLD Data Base

5. FUNDING NUMBERS
Contract No.
N61339-91-D-0001

6. AUTHOR(S)
Compiled by: Brett Butler
Tony Valle
Jim Watson

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Loral Systems Company
ADST Program Office
12151-A Research Parkway
Orlando, FL 32826-3283

8. PERFORMING ORGANIZATION REPORT NUMBER
ADST-94-W003283

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Simulation, Training and Instrumentation Command (STRICOM)
12350 Research Parkway
Orlando, FL 32826-3275

10. SPONSORING ORGANIZATION REPORT

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE
A

13. ABSTRACT (Maximum 200 words)

This document describes the status of a prototype implementation of the SIMWORLD Data Base (SWDB). The SIMWORLD Data Base is the major component of the Common Data Base (CDB), which is a key element of the BDS-D Architecture concept (Version 1.0). The document concerns the establishment of an ORACLE relational data base with an initial complement of DIS entities and assets structured in compliance with a proposed Common Data Base Standard. The development of the data base structure and evaluation of an interface to a DIS SAF simulator (ModSAF) are described. Recommendations for substantive modifications to the proposed CDB Standard are made. Aspects of the next incremental extension of the CDB concept toward supporting an exercise planning capability are also discussed.

14. SUBJECT TERMS
ADST, BDS-D, Common Data Base, SIMWORLD Data Base, STRICOM, Systems Engineering

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	**18. SECURITY CLASSIFICATION OF THIS PAGE**	**19. SECURITY CLASSIFICATION OF ABSTRACT**	**20. LIMITATION OF ABSTRACT**
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

Table of Contents

List of Figures

List of Tables

Abstract

This paper describes the status of a prototype implementation of the SIMWORLD Data Base (SWDB). The SIMWORLD Database is the major component of the Common Data Base (CDB) which is a key element of the BDS-D Architecture concept (Version 1.0) The work reported here is the establishment of an ORACLE relational data base with an initial complement of DIS entities and assets structured in compliance with a proposed Common Data Base Standard. The development of the data base structure and evaluation of an interface to a DIS SAF simulator (MODSAF) are described. Recommendations for substantive modifications to the proposed CDB Standard are made to remove recursive features and generally facilitate the interface with object oriented DIS simulators. This is the initial increment of a spiral development effort directed at achieving long term DIS interoperability objectives while obtaining near term confirmation of the approach through utility demonstrations. Aspects of the next incremental extension of the CDB concept toward supporting an exercise planning capability are also discussed. This work is being performed as a part of the U.S. Army STRICOM BDS-D Architecture and Standards effort, by SPARTA, Inc. under subcontract to Loral Advanced Distributed Simulation.

1 Introduction.

The standardization of DIS message protocols in the form of protocol data units (PDUs) was an important initial step in realizing the potential benefit of DIS for interactive military training. Standardized databases with a common set of public information shared between networked participants are equally important to achieving a correlated view of the virtual environment and interoperability on a heterogeneous simulation network. This operational concept is diagrammed in Figure1-1. The BDS-D architecture approach defines the composite set of DIS public information databases (virtual and physical) required to achieve interoperability to be the SIMWORLD database. The structure and meaning of information collected in the SWDB is standardized for common understanding and shared usage by the Common Data Base Standard (ADST/WDL/TR-93-W003007, Reference 1). This report provides the initial status of a prototype implementation of the SWDB concept and recommendations for substantive improvements to the Common Data Base Standard.

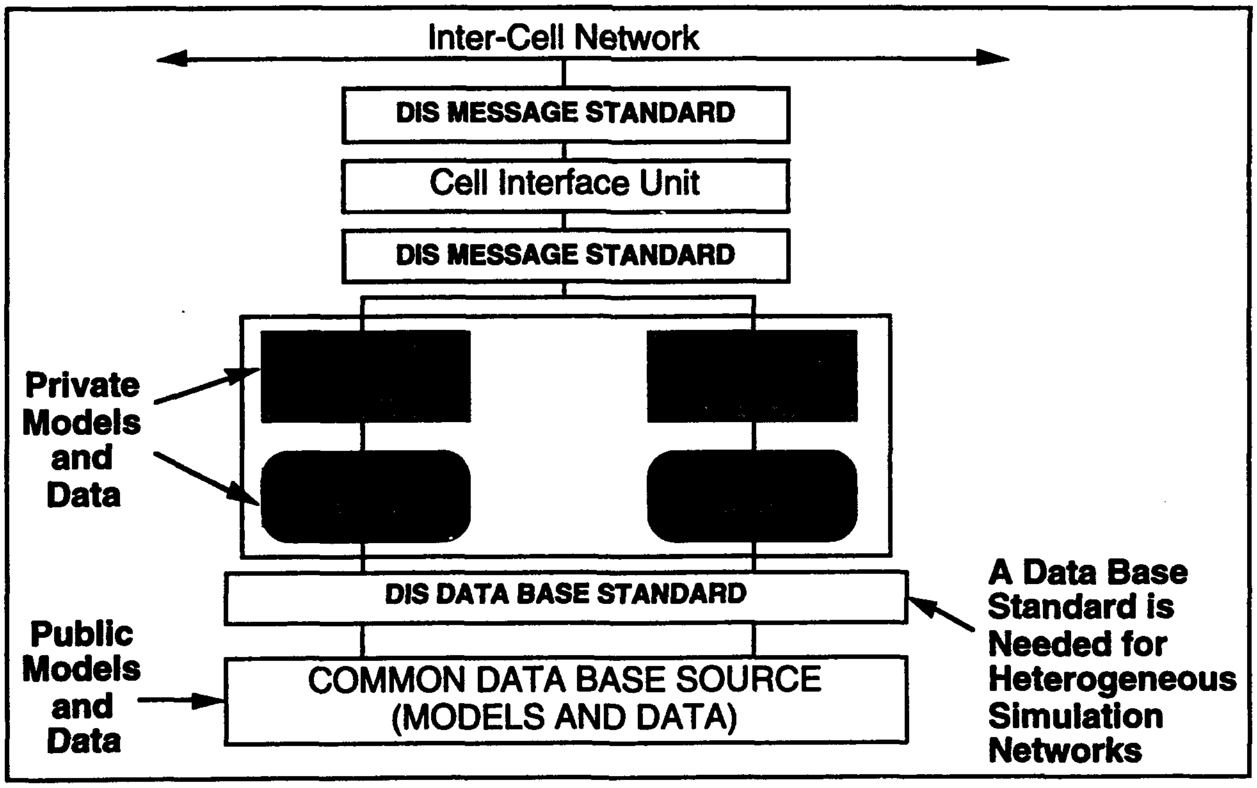

Figure 1-1. Proposed DIS Architecture

A subset of database elements required for a specific exercise is drawn from the SWDB to form a Session Data Base (SDB). Results of the executed exercise are compiled in a Review Data Base (RDB). The SDB and RDB also become elements of the SWDB for future reference or use as shown in Figure 1-2. Elements of the virtual world are referred to as entities, elements of the physical world required to implement the virtual world are referred to as assets; this is clarified in the DIS context in Figure 1-3. Section 3.0 of this report provides an evaluation and status of the initial entity and asset population of the prototype ORACLE relational database. Early experience with implementation concepts for an operational interface between the SWDB and a simulator (ModSAF) data base to be used in subsequent SDB and RDB demonstrations is described in Section 4.0.

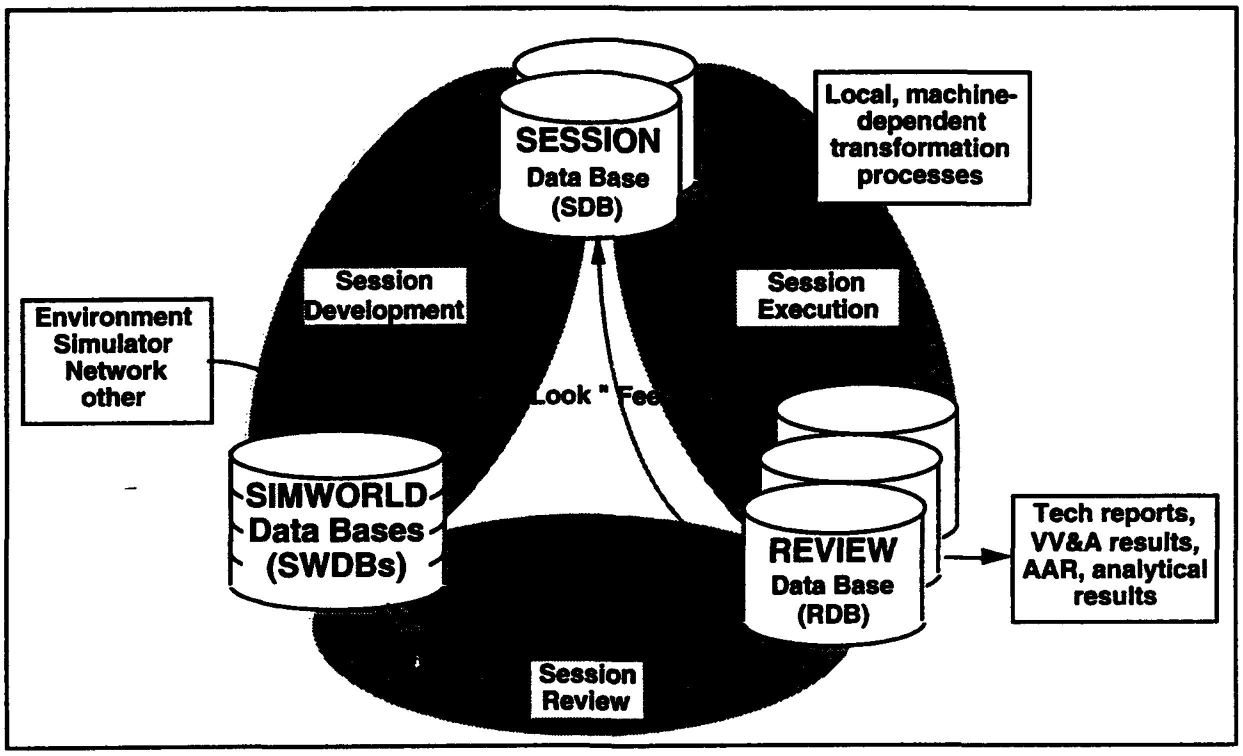
SESSION
Data Base
(SDB)
Local, machine-dependent transformation processes
Session Development
Session Execution
Environment Simulator Network other
Look " Fee
SIMWORLD Data Bases (SWDBs)
REVIEW
Data Base
(RDB)
Tech reports, VV&A results, AAR, analytical results
Session Review

Figure 1-2. DIS Architecture Common Data Base (CDB)

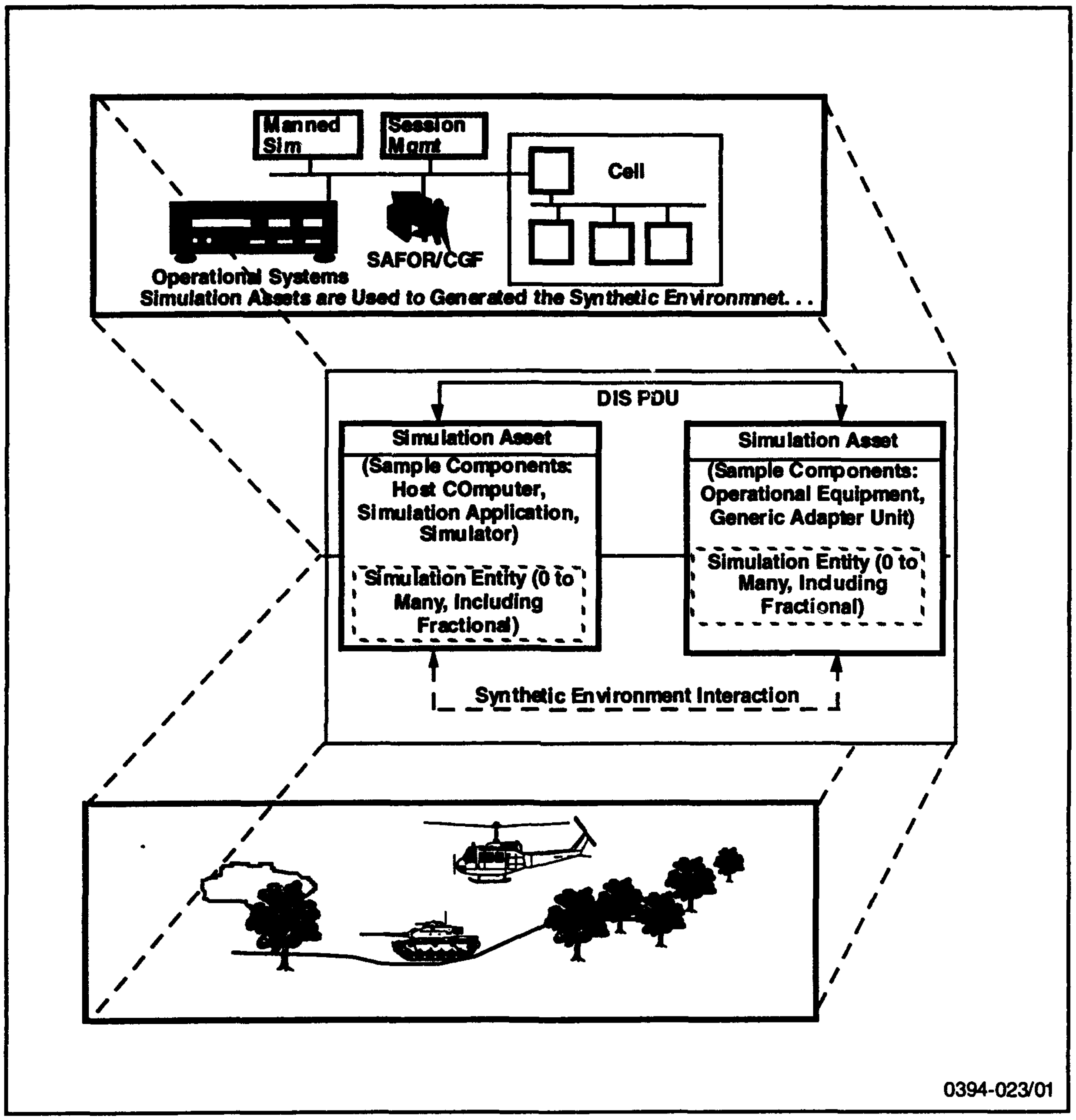
Manned Sim
Session Mgmt
Cell
SAFOR/CGF
Operational Systems
Simulation Assets are Used to Generated the Synthetic Environmnet. . .
DIS PDU
Simulation Asset
(Sample Components: Host COmputer, Simulation Application, Simulator)
Simulation Entity (0 to Many, Including Fractional)
Simulation Asset
(Sample Components: Operational Equipment, Generic Adapter Unit)
Simulation Entity (0 to Many, Including Fractional)
Synthetic Environment Interaction
0394-023/01

Figure 1-3. Simulation Entities and Assets

The goal for the SWDB development program is to create a database utilizing the Common Database Standard that will contain everything necessary to run a DIS exercise and a capability (via simulation) to manage and preview DIS configurations and exercises. This capability would be designed to support the following functions: Planning and scheduling DIS exercises (simulator scheduling conflicts, ...), evaluation of exercise resource capability (network traffic, ...), preview ability of exercise configuration to meet exercise goals (do the elements and tactics selected interact adequately to meet the objectives of the exercise), etc.

The next incremental capability demonstration which would allow the user to open additional windows in ModSAF listing entities/assets for Ft. Knox and Ft. Rucker is briefly outlined in Section 4. The capability to examine entity/asset characteristics at each location and/or select and place an entity into a simulation scenario is also planned. These features will enable subsequent development of exercise planning, execution and review required for testing the full CDB Standard concept.

2 SWDB Requirements.

As defined in ADST/WDL/TR-93-W003007, the Common Data Base (CDB) is a collection of machine-independent databases that supports interoperable DIS operations in a heterogeneous simulation network. The CDB will contain three component data bases: the SIMWORLD Data Base (SWDB), the Session Data Base (SDB), and the Review Data Base (RDB).

The SIMWORLD Data Base (SWDB) is the principal component data base of the CDB and will be a standard, configuration managed library of machine-independent information which is structured in the form of types of simulation entities and simulation assets. In practice, the simulation entities and simulation assets within a SWDB will meet defined interoperability criteria.

The principal functional requirements of the SWDB are illustrated in Figure 2-1. The composite capabilities required include: (1) data extraction from source data bases, (2) data analysis and validation against acceptance criteria, (3) data reformatting to a standard interchange format, (4) data storage, maintenance and configuration management, (5) data acquisition support to data base users, (6) data distribution to data base users.

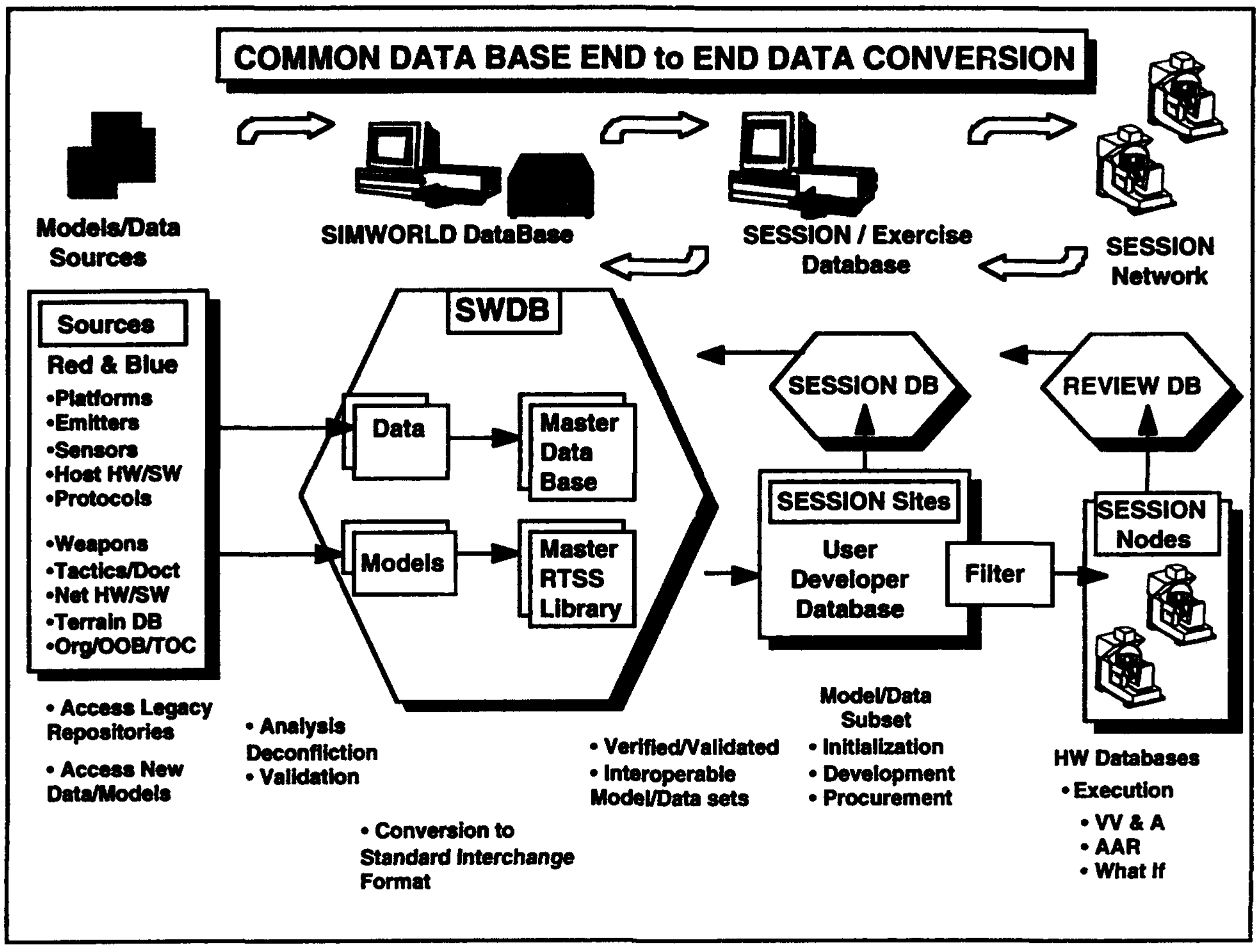

Figure 2-1. DIS Common Data Base End to End Data Conversion

The SWDB and other component data bases will be logically constructed according to a Common Data Base Standard. The contents of the databases are defined in terms of a Common Data Base Interchange Format (CDBIF) which will facilitate interchange and correlated, interoperable usage. The CDB standard will also define the mandatory and optional contents of the component data bases and may define procedures by which these databases are to be utilized.

The CDBIF will use a formal, widely used language structure. The CDBIF will allow machine-independent exchange of data to support DIS operations.

3 Current Implementation -- Database.

The initial capability increment is focused on defining a highly effective and efficient operational data base standard structure. This addresses functional requirements (1), (3), (4) and (6) identified in Figure 2-1.

3.1 SIMWORLD Database Structure.

The primary design criterion for implementing the initial prototype SIMWORLD database was conformance to the Distributed Interactive Simulation Common Database Standard. The structure for this database is contained in ADST/WDL/TR-93-W003007 (Reference 1). A brief description of this standard is necessary to understand the implementation decisions made in the ORACLE database.

3.1.1 Books and Chapters.

The Common Database (CDB) Standard specifies the structure of entity and asset data for SIMWORLD. It is organized as a stream of structured text which is divided into six books. Book1 contains global information about the SIMWORLD database, security levels, and version information. Book2 contains a list of the entities and assets contained in the database. Book3 details the entities and their associated data. Book4 details all the information for assets. Book5 is divided into two chapters, with the first listing definitions for enumerated data elements and the second providing the values for included data tables. Book 6 is currently unimplemented. The collection of all six books and any associated external files, such as terrain files, constitutes a SIMWORLD database.

3.1.2 Hierarchical Structure.

The CDB standard specifies a hierarchical format for the data designed to minimize the problems of transporting data from one machine and implementation to another. The format is constructed so as to permit a simple parser to interpret the data stream. Each book, chapter, and subsection consists of a header line, a body part, and a trailer line. The body part may, itself, contain further structure organized recursively.

3.1.3 Information Detail.

All the detailed numerical and enumerated data in the CDB standard is organized into two structures: data tables and enumerated tables. The format for these tables is restrictive and inflexible. All the data for any entity or asset must be collected into one of these forms. In addition, entities are defined with five subparts: capabilities, susceptibilities, appearances, tactics, and environments. All data items for entities must be associated with one of these five categories. The specification for the CDB text format is in a modified Backus-Naur format (BNF) with optional, repeating, and nested text descriptions organized around keywords and end-of-line markers.

3.1.4 Design Philosophy.

SIMWORLD is intended to support the development of the Distributed Interactive Simulation (DIS) network. This is a heterogeneous network which will ultimately host hundreds of machines interacting simultaneously, generally in real-time. As a result, the CDB standard has been constructed with portability as a key requirement. The result is an ASCII text representation for data with a machine-parsable structure. There is no requirement within the CDB standard that the internal representation of the data must conform to this text-based format. In the development of our prototype SIMWORLD database, we chose to implement in ORACLE a structure which was closely analogous to the BNF text description of the database. This was done to guarantee that we had a good understanding of the detailed CDB structure and to ease the data-entry burden.

3.2 ORACLE Implementation Structure.

3.2.1 Relational Structure.

ORACLE is a relational database system which means that data is stored in the form of rows within tables. Tables contain a fixed set of columns, each column having a fixed length, data type, and possibly other attributes (such as uniqueness or a requirement to not be null). Relational databases are very flexible but require an experienced designer to implement complex data structures. In particular, hierarchical data can be represented in a relational database, but it presents a unique set of design difficulties and can have severe impacts on the efficiency of the database engine. In order to delineate specific problem areas, we elected to implement the hierarchical CDB database format directly in ORACLE. This entails the use of multiple tables to represent each complex data item in the standard. Each optional or repeating data item must be set off in a separate table to make sure that there is no duplication of data or excessive numbers of nulls. In short, the richer the structure, the more tables are necessary (in general) to represent the data.

3.2.2 Table Details.

Our implementation of the CDB database is in third normal form and involves 46 tables, 10 of which are "crossover" tables used to implement many-to-many relationships between the underlying constructs. Each major BNF element in the CDB standard is represented as a field in one of these tables. Collections of such fields are organized into tables which permit "one or more" instances of such items to appear. The tables are linked through the use of unique sequence numbers. Walking through a single example will illustrate the design of the entire structure. We will consider the implementation of the data tables as outlined in section 4.3.5.2.1 of the CDB standard. The relevant BNF text is as follows (Figure 3-1):

```
DATA_TABLE <Data_Table_Name>
NUM_DIMENSIONS <dimensions>
(DIMENSION <dimension_number> <Dimension_Name> (<Uom>)...
DATA_TABLE_VALUES (<Dimension_Name>)...
<dimension_value> or <dimension_range>)...
END  DATA_TABLE_VALUES
END DATA_TABLE <Data_Table_Name>
```

Figure 3-1. Relevant BNF Text

The implementation of this consists of four linked tables defined as follows in Table 3-1:

Table 3-1. Implementation of Relevant BNF Text

DATA_TAB	
DT_ID	Sequence
DT_Name	name
DATA_DIMS	
O_DT_ID	Owner ID
DD_Dim_Num	Dimension number
DD_Dependent?	Dependent variable?
DD_Name	name
DD_Units	units
DATA_ENTRIES	
O_DT_ID	Owner ID
DE_Entry	Entry number
DE_Dim_Num	Dimension number
DE_Min	minimum range
DE_Max	maximum range
DATA_VALUES	
O_DT_ID	Owner ID
O_DE_Entry	Owner entry number
DV_Dim_Num	Dimension number
DV_Value	table value

The data table name and its unique ID number are stored in the DATA_TAB table. Each dimension within that table gets a row in the DATA_DIMS table which is linked to the DATA_TAB table through the O_DT_ID field — a foreign key value. Each dimension has a unique dimension number (DD_Dim_Num) which is used to collate entries and to order the values in the table. The dimension's name, unit of measure, and a Boolean indicating whether the dimension is independent or not is also contained in this table.

Each table consists of a number of entries. An entry should be thought of as a cell in the multi-dimensional space of the independent variables associated with the table. That is, a particular entry consists of range limits for each possible independent variable. The DATA_ENTRIES table contains a row for each independent variable range for each entry in the data table.

For example, let us suppose that the data table **Cross-Section** contains radar cross-section information for all azimuth and elevation angles. This is a three-dimensional table with independent variables **azimuth** and **elevation** and dependent variable **sigma**. If the cross-section is specified for each 5 degrees of azimuth and each 2 degrees of elevation, we would expect the table to contain 6,480 entries. Consequently, the DATA_ENTRIES table would contain twice that, or 12,960 rows, each row having a limit on the range either **azimuth** or **elevation**. The first few rows in the table might look like those in Table 3-2 below:

Table 3-2. DATA ENTRIES Table Example

	O_DT_ID	DE_Entry	DE_Dim_Num	DE_Min	DE_Max
row 1	45	1	1	0	5
row 2	45	1	2	0	2
row 3	45	2	1	5	10
row 4	45	2	2	0	2

Each entry has one or more values associated with it — one for each of the dependent variables. The DATA_VALUES table contains these numbers, tied to the corresponding entry through the O_DE_Entry field. Similar constructs exist throughout the design. Each entity can have several capabilities, for example, so the table E_C_X contains the ID of the entity and its associated capabilities. Each capability, in turn, may contain data tables. This relationship is captured in the C_DT_X table linking capability IDs and the associated data table IDs. The naming conventions for the tables and fields are consistent throughout the database and can be followed easily. Appendix A contains a full description of the columns and tables for the entire database.

3.2.3 CDB Prototype.

Since the data structure for the ORACLE implementation closely matches the CDB format, exporting a CDB-compliant database is a simple matter. A set of reports named book1.sql, book2.sql, etc., have been devel[illegible]ed which extract the relevant information from the ORACLE system and create the corresponding books from the CDB standard. These reports use the PL/SQL language which comes with the procedural option of the ORACLE system. The reports produce a set of files named BOOK1.lst, BOOK2.lst, etc., which can be concatenated to yield a complete CDB database. The complete text of the reports is presented in Appendix B.

In order to test the implementation, we chose the ModSAF program — a semi-automated force simulator which runs on Sun and Silicon Graphics workstations. ModSAF comes with an extensive and well-structured library of entities. The ORACLE SIMWORLD prototype consists of three entities translated from the ModSAF database: an AH-64 Apache helicopter and two tanks, an M1A1 and a T-72. Together, these three entities resulted in the creation of 38 sub-entities, 60 enumerated tables and over 120 data tables. The database contains all the tables, fields, and values contained in ModSAF which are required to implement these entities.

3.3 Problems.

A number of problems with the current CDB Standard became evident during the course of this task. The structure, while easily machine parsed, is too rigid to accommodate many common constructs from the ModSAF database. This lack of flexibility led to difficulties in assigning ModSAF to appropriate locations within the CDB structure, as well as the dissociation of data elements which were represented as a unit within ModSAF. Our judgment is that this would lead to long-term difficulties in trying to extract and collate information from the CDB format which would support a wide range of computer simulations.

3.3.1 Attribute Data.

Computer constructs often contain data which is associated with an entity for the simulation's use only and do not necessarily correspond to real-world information. Such attribute data does fit into the CDB model, which requires all entity data to be associated with a capability, susceptibility, appearance, tactic, or environment. This makes information such as icons, screen colors, data sizes, record types, and ID numbers difficult or impossible to assign properly to the entity. While it is possible to force the data into categories that may not be completely appropriate, or to create new entities which encapsulate the data specific to a particular simulation, this defeats the purpose of a single common database standard: the free exchange of information between different simulation programs.

3.3.2 Data Cohesion.

Often, we came across grouped ModSAF data which contained pieces that belonged to separate sub-parts of the CDB standard. Breaking this data up would permit it all to be included in the standard, but greatly exacerbates the problem of extracting a ModSAF-readable database from the ORACLE system. In fact, this was probably the single most troublesome difficulty encountered in the CDB standard and can be traced back to its hierarchical structure. As more simulations are added to the database, this would almost certainly become a major problem that would threaten the utility of the common database itself. It is essential that related data retain their associations so that the simulators can find all the information they need to construct an entity.

3.3.3 Structure Flexibility.

The foundational elements in the CDB standard are data tables and enumerated tables. The data tables are designed to handle hyper-rectangular arrays of numerical information and enumerated tables are designed to associate text meaning with numerical field values. There is really no construct for a complex array, a collection of related non-numeric objects, or a dictionary associating keys and values. This lack of flexibility resulted in extensive work-arounds and "force-fitting" of the ModSAF constructs into the CDB standard. The database would be greatly simplified by the addition of these kinds of collections and a freer format for data records and groups.

3.3.4 Correlations.

The worst sin in the CDB standard is the assumption that all correlations between data elements can be fitted into the hierarchy defined in the standards document. This is a reasonable assumption on its face, but rapidly breaks down when applied to real-world data. For instance, one ModSAF construct shows that the M1A1 and the T-72 are lined in the sense that they share an internal representation area and some simulation-specific attributes. This can be handled through the use of common appearances and capabilities (which is, in fact, how the data was entered) but that is a poor fit to the actual relationship. What is actually going on here is that the M1A1 and the T-72 are both instances of a common class — Tank — which cannot be well-represented in the CDB format.

3.4 Recommendations.

From our extensive work with the ModSAF data and the CDB standard, we think that some changes to the standard would greatly improve its utility and that they could be accomplished at this early stage with a minimum of disruptive effects and for a reasonable cost. It is a truism that extra effort expended to ensure a good product in the design phase of any information system will be recouped many times over in the effort of implementation and maintenance.

3.4.1 Object-Oriented Structure.

The real world is a collection of objects. Modern languages, databases, and design tools exist to develop and capture this object structure in an information system. The ModSAF database itself is an excellent example of proper object-oriented design. Rather than a collection of fixed structures, ModSAF uses a flexible language with inheritance, substitution, and extension to implement its data sets. This sort of structure is consistent with modern development in simulators and database systems. It represents an efficient means of associating related data, avoids duplication, and minimizes the amount of traffic required to access or synchronize data over a network.

We recommend that the CDB standard be altered to accommodate an object-based structure. This would involve the development of a class hierarchy appropriate for simulators which at least captures the level of detail in the current standard. Actually, that is not difficult — one can, in fact, produce a more detailed class hierarchy than the current standard very easily. The class hierarchy would include mandatory attributes and specify inheritance relationships for each class of interest. The attributes would correspond to the current "required fields" in the CDB standard. Since all object structures share the common features of extensibility, data abstraction, and data encapsulation, they are ideal for implementation of common "pools" of information such as the CDB requires.

In addition to the class hierarchy, an entity-relationship (ER) structure for the CDB should be developed. This would entail identifying the most likely associations and ownership relations expected in the CDB environment. In other words, it would establish a basis for data structures which capture information like "this main gun belongs to this tank" and "all fixed-wing aircraft have a stall speed". Like a class hierarchy, an ER diagram can be altered and extended to include new developments and more information so the system need not become obsolete, nor must the designer anticipate all possible associations in the preliminary design.

3.4.2 Data Structure.

For object-oriented systems, a rich class structure of data types has been in use for several years: the data classes of the language Smalltalk. These classes permit arbitrarily complex data associations and structures including arrays, tables, dictionaries, sets, and ordered collections. By including this rich data representation set in the CDB standard, it will possible for simulations to keep associated data in close proximity and still allow "foreign" systems to access that data in a meaningful way. In addition, items that must be restricted for security reasons are much easier to isolate by using such a system. Individual data items or collections can be tagged with classification levels and efficiently hidden from access as required.

3.4.3 Cross-Platform Compatibility.

The object structure outlined above can be implemented in almost any database system on any platform. Text representations for arbitrarily complex objects can be generated (another problem that has been solved in Smalltalk for over a decade) and shipped across platforms just as for the CDB standard. Parsing an object structure is actually easier than parsing a hierarchical structure because the stream has little or no "state information" that must be tracked.

4 Current Implementation -- ModSAF Utilization.

The computer generated forces model, ModSAF, was provided to SPARTA by LORAL to facilitate a prototype demonstration of the SWDB concept. The following is a summary of the progress to date in implementing ModSAF on Silicon Graphics and Sun SPARC stations in the SPARTA, Huntsville facility.

An early requirement in the use of ModSAF was to access the ModSAF entity databases to extract data to be inserted into the ORACLE database implementation discussed above. A C program was created for the conversion of the ModSAF database to a format compatible with ORACLE. This involved extracting the filereader from ModSAF in order to read the databases. The program currently has a simple interface with the user and this program will be further modified once the exact format for the output is obtained.

ModSAF has been successfully built on an SGI and a Sparc 10 (using GNU GCC 2.5.7). We have successfully run ModSAF under a single host configuration (SAFstation and SAFsim on the same machine). This was accomplished on the SGI as well as the Sparc 10.

Several scenarios have been run to investigate the SAFstation and SAFsim operations (creating and controlling units, assigning missions, the various editors, etc.). We have also successfully distributed ModSAF over our local network and have run a SAFstation along with up to four SAFsims using Sparc 10's and Sparc 2's. All of the stations and simulators were on the same exercise and database. We have also successfully run two SAFstations and up to three SAFsims on our local network with the SAFstations utilizing independent PO databases and the SAFsims divided among these databases. We have attempted a distributed run involving a Sparc 10 and a SGI platform, but experienced problems with the platforms communicating with each other.

The Logger has been built on a Sparc 10 and we have successfully run an exercise utilizing the Logger. The Logger will be utilized to playback previously run exercises to highlight differences in performance for a given set of entity data extracted from the SWDB.

Appendix A — Table Descriptions

```
APPEARANCES
 Name                            Null?    Type
 ------------------------------- -------- ----
 AP_ID                                    NUMBER
 AP_TYPE                                  CHAR(30)
 AP_DESCRIPTION                           CHAR(80)
 COMMON                                   CHAR(1)

APP_EXTENTS
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_AP_ID                                  NUMBER
 APP_EXTENT                               CHAR(30)

APP_STDS
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_AP_ID                                  NUMBER
 AP_STANDARD                              CHAR(30)
 AP_VERSION                               CHAR(30)
 AP_INFO                                  CHAR(80)

AP_DT_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_AP_ID                                  NUMBER
 C_DT_ID                                  NUMBER

ASSETS
 Name                            Null?    Type
 ------------------------------- -------- ----
 A_ID                                     NUMBER
 A_NAME                                   CHAR(30)
 A_TYPE                                   CHAR(30)
 A_IDENTITY                               CHAR(80)

CAPABILITIES
 Name                            Null?    Type
 ------------------------------- -------- ----
 C_ID                                     NUMBER
 C_TYPE                                   CHAR(30)
 C_DESCRIPTION                            CHAR(80)
 COMMON                                   CHAR(1)

CONTAINS
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_E_ID                                   NUMBER
 C_E_ID                                   NUMBER

CON_EXCEPTS
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_E_ID                                   NUMBER
 C_E_ID                                   NUMBER
 CE_TYPE                                  CHAR(1)
 CE_CLASS                                 CHAR(18)
 CE_ID                                    NUMBER

C_DT_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_C_ID                                   NUMBER
 C_DT_ID                                  NUMBER

DATA_DIMS
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_DT_ID                                  NUMBER
 DD_DIM_NUM                               NUMBER
 DD_DEPENDENT                             CHAR(1)
 DD_NAME                                  CHAR(30)
 DD_UNITS                                 CHAR(18)

DATA_ENTRIES
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_DT_ID                                  NUMBER
 DE_ENTRY                                 NUMBER
 DE_DIM_NUM                               NUMBER
 DE_MIN                                   NUMBER
 DE_MAX                                   NUMBER

DATA_TAB
 Name                            Null?    Type
 ------------------------------- -------- ----
 DT_ID                                    NUMBER
 DT_NAME                                  CHAR(30)

DATA_VALUES
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_DT_ID                                  NUMBER
 O_DE_ENTRY                               NUMBER
 DV_DIM_NUM                               NUMBER
 DV_VALUE                                 NUMBER

DIS_PDU_TYPES
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_DP_ID                                  NUMBER
 DPT_NAME                                 CHAR(30)
 DPT_SR                                   CHAR(1)
 DPT_RATE                                 NUMBER
 DPT_KIND                                 CHAR(1)

DIS_PROTOCOLS
 Name                            Null?    Type
 ------------------------------- -------- ----
 DP_ID                                    NUMBER
 O_A_ID                                   NUMBER
 DP_TYPE                                  CHAR(30)
 DP_VERSION                               CHAR(30)
 DP_OOP                                   CHAR(1)
 DP_ERROR                                 CHAR(1)
 DP_PDU_IMP                               CHAR(30)

DT_ET_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 DT_ID                                    NUMBER
 ET_ID                                    NUMBER

ENTITIES
 Name                            Null?    Type
 ------------------------------- -------- ----
 E_ID                                     NUMBER
 E_NAME                                   CHAR(30)
 E_PUBLIC                                 CHAR(1)
 E_IDENTITY                               CHAR(255)
```

```
ENUM_FIELDS
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_ET_ID                                  NUMBER
 EF_FIELD_NUM                             NUMBER
 EF_NAME                                  CHAR(30)
 EF_UNITS                                 CHAR(18)
 EF_FIELD_SIZE                            CHAR(12)

ENUM_TAB
 Name                            Null?    Type
 ------------------------------- -------- ----
 ET_ID                                    NUMBER
 ET_NAME                                  CHAR(30)
 ET_ITEM_SIZE                             NUMBER

ENUM_USES
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_ET_ID                                  NUMBER
 EU_USER_PDU                              CHAR(30)
 EU_USER_RECORD                           CHAR(30)

ENUM_VALUES
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_ET_ID                                  NUMBER
 EV_FIELD_NUM                             NUMBER
 EV_MIN                                   NUMBER
 EV_MAX                                   NUMBER
 EV_MEANING                               CHAR(80)

ENVIRONMENTS
 Name                            Null?    Type
 ------------------------------- -------- ----
 ENV_ID                                   NUMBER
 ENV_TYPE                                 CHAR(30)
 ENV_DESCRIPTION                          CHAR(80)
 COMMON                                   CHAR(1)

ENV_DT_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_ENV_ID                                 NUMBER
 C_DT_ID                                  NUMBER

ENV_STDS
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_ENV_ID                                 NUMBER
 ENV_STANDARD                             CHAR(30)
 ENV_VERSION                              CHAR(30)
 ENV_INFO                                 CHAR(80)

E_A_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_E_ID                                   NUMBER
 C_A_ID                                   NUMBER

E_C_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_E_ID                                   NUMBER
 C_C_ID                                   NUMBER
```

```
E_ENV_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_E_ID                                   NUMBER
 C_ENV_ID                                 NUMBER

E_S_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_E_ID                                   NUMBER
 C_S_ID                                   NUMBER

E_T_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_E_ID                                   NUMBER
 C_T_ID                                   NUMBER

HARDWARE
 Name                            Null?    Type
 ------------------------------- -------- ----
 HW_ID                                    NUMBER
 O_A_ID                                   NUMBER
 HW_NAME                                  CHAR(30)
 HW_TYPE                                  CHAR(30)
 HW_IDENT                                 CHAR(80)

HW_DETAIL
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_HW_ID                                  NUMBER
 HWD_TYPE                                 CHAR(30)
 HWD_DESCRIPTION                          CHAR(80)

IF_DETAIL
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_IF_ID                                  NUMBER
 DF_NAME                                  CHAR(80)
 DF_SIZE                                  NUMBER

IF_SUPPORT
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_IF_ID                                  NUMBER
 IFS_TYPE                                 CHAR(18)
 IFS_DESCRIPTION                          CHAR(80)

IMAGES
 Name                            Null?    Type
 ------------------------------- -------- ----
 LINE_NO                                  NUMBER
 LINE                                     CHAR(255)

INCLUDE_FILES
 Name                            Null?    Type
 ------------------------------- -------- ----
 IF_ID                                    NUMBER
 O_A_ID                                   NUMBER
 IF_TYPE                                  CHAR(30)
 IF_CONTENT                               CHAR(80)
 IF_FILE_TYPE                             CHAR(12)
```

```
ISO_PROTOCOLS
 Name                            Null?    Type
 ------------------------------- -------- ----
 IP_ID                                    NUMBER
 O_A_ID                                   NUMBER
 IP_LAYER                                 NUMBER
 IP_PROTOCOL                              CHAR(30)
 IP_LAYER_NAME                            CHAR(12)
 IP_PDU_NAME                              CHAR(30)
 IP_SERVICE                               CHAR(80)

PERSONNEL
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_A_ID                                   NUMBER
 P_TYPE                                   CHAR(30)
 P_DESCRIPTION                            CHAR(80)
 P_QUALIFICATIONS                         CHAR(80)

SOFTWARE
 Name                            Null?    Type
 ------------------------------- -------- ----
 SW_ID                                    NUMBER
 O_A_ID                                   NUMBER
 SW_NAME                                  CHAR(30)
 SW_TYPE                                  CHAR(30)
 SW_BUILD_DATE                            DATE

SUPPORTS
 Name                            Null?    Type
 ------------------------------- -------- ----
 SU_ID                                    NUMBER
 O_A_ID                                   NUMBER
 SU_MAX_OWN                               NUMBER
 SU_MAX_TRACKED                           NUMBER

SUSCEPTIBILITIES
 Name                            Null?    Type
 ------------------------------- -------- ----
 S_ID                                     NUMBER
 S_TYPE                                   CHAR(30)
 S_DESCRIPTION                            CHAR(80)
 COMMON                                   CHAR(1)

SU_CONC_ET
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_SU_ID                                  NUMBER
 SU_EN_NAME                               CHAR(30)

SWDB_SEC_MODS
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_SWDB_ID                                NUMBER
 SEC_SECTION                              NUMBER
 SEC_BASE                                 CHAR(1)
 SEC_CLASS                                CHAR(30)

SWDB_TAB
 Name                            Null?    Type
 ------------------------------- -------- ----
 SWDB_ID                                  NUMBER
 SWDB_SECURITY                            CHAR(18)
 CDB_VERSION_NUM                          NUMBER
 SWDB_NAME                                CHAR(30)
 SWDB_VERSION                             NUMBER
 SWDB_PURPOSE                             CHAR(80)
 SWDB_POC                                 CHAR(255)

SW_DETAIL
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_SW_ID                                  NUMBER
 SWD_TYPE                                 CHAR(30)
 SWD_DESCRIPTION                          CHAR(80)

S_DT_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_S_ID                                   NUMBER
 C_DT_ID                                  NUMBER

TACTICS
 Name                            Null?    Type
 ------------------------------- -------- ----
 T_ID                                     NUMBER
 T_TYPE                                   CHAR(30)
 T_DESCRIPTION                            CHAR(80)
 COMMON                                   CHAR(1)

T_DT_X
 Name                            Null?    Type
 ------------------------------- -------- ----
 O_T_ID                                   NUMBER
 C_DT_ID                                  NUMBER
```

Appendix B — Reports

```
set head off;
set pages 0;
set feedback off;
spool BOOK1;
select 'SWDB_BOOK1' from dual;
select 'SWDB_SECURITY_LEVEL '||SWDB_Security from swdb_tab;
select sec_class from swdb_sec_mods where sec_section =1 and sec_base = 'n';
select 'CDB_VERSION_NUMBER '||CDB_Version_Num from swdb_tab;
select 'SIMWORLD '||SWDB_Name||' '||SWDB_Version from swdb_tab;
select 'SIMWORLD_PURPOSE '||SWDB_Purpose from swdb_tab;
select 'SIMWORLD_POC '||SWDB_POC from swdb_tab;
select 'NUMBER_ENTITY_TYPES '||count(distinct E_ID) from entities;
select 'NUMBER_ASSET_TYPES '||count(distinct A_ID) from assets;
select 'END SWDB_BOOK1' from dual;
spool off;
set head off;


set pages 0;
set feedback off;
spool BOOK2;
select 'SWDB_BOOK2' from dual;
select 'SWDB_BOOK2_SECURITY_LEVEL '|| sec_class
   from swdb_sec_mods where sec_section =2 and sec_base = 'y';
select sec_class from swdb_sec_mods where sec_section =2 and sec_base = 'n';
select 'ENTITY_TYPE '||E_ID||' '||E_Name from entities;
select 'ASSET_TYPE '||A_ID||' '||A_Name from assets;
select 'END SWDB_BOOK2' from dual;
spool off;


set head off;
set pages 0;
set feedback off;
spool BOOK3
select 'SWDB_BOOK3' from dual;
select 'SWDB_BOOK3_SECURITY_LEVEL '|| sec_class
   from swdb_sec_mods where sec_section =3 and sec_base = 'y';
select sec_class from swdb_sec_mods where sec_section =3 and sec_base = 'n';
delete from images;

DECLARE
  lino number := 0;
  line char(255);
  eid number;
  cid number;
  cursor cEntity is
    select E_ID, E_Name, E_Public, E_Identity
    from entities;

  cursor cCapability is
    select C_ID, C_Type, C_Description
    from capabilities, E_C_X
    where o_e_id = eid
    and C_C_ID = C_ID;
  cursor cDT_cap is
    select DT_Name
    from data_tab, C_DT_X
    where O_C_ID = cid
    and DT_ID = C_DT_ID;

  cursor cSusc is
    select S_ID, S_Type, S_Description
    from susceptibilities, E_S_X
    where o_e_id = eid
    and C_S_ID = S_ID;
  cursor cDT_sus is
    select DT_Name
    from data_tab, S_DT_X
    where O_S_ID = cid
    and DT_ID = C_DT_ID;

  cursor cAppearance is
    select AP_ID, AP_Type, AP_Description
    from appearances, E_A_X
    where o_e_id = eid
    and C_A_ID = AP_ID;
  cursor cDT_app is
    select DT_Name
    from data_tab, AP_DT_X
    where O_AP_ID = cid
    and DT_ID = C_DT_ID;

  cursor cEnvironment is
    select ENV_ID, ENV_Type, ENV_Description
    from environments, E_ENV_X
    where o_e_id = eid
    and C_ENV_ID = ENV_ID;
  cursor cDT_env is
    select DT_Name
    from data_tab, ENV_DT_X
    where O_ENV_ID = cid
    and DT_ID = C_DT_ID;

  cursor cTactic is
    select T_ID, T_Type, T_Description
    from tactics, E_T_X
    where o_e_id = eid
    and C_T_ID = T_ID;
  cursor cDT_tac is
    select DT_Name
    from data_tab, T_DT_X
    where O_T_ID = cid
    and DT_ID = C_DT_ID;

  cursor cContains is
    select E_Name, C_E_ID
    from Entities, Contains
    where O_E_ID = eid
    and E_ID = C_E_ID;
  cursor cConex is
    select CE_Type, CE_Class, CE_ID
    from CON_EXCEPTS
    where O_E_ID = eid
    and C_E_ID = cid
    order by CE_Type, CE_Class;

  cursor cComCap is
    select C_ID, C_Type, C_Description
    from capabilities
    where common = 'y';

  cursor cComSusc is
    select S_ID, S_Type, S_Description
    from susceptibilities
    where common = 'y';

  cursor cComApp is
    select AP_ID, AP_Type, AP_Description
    from appearances
    where common='y';

  cursor cComEnv is
    select ENV_ID, ENV_Type, ENV_Description
    from environments
    where common='y';

  cursor cComTac is
    select T_ID, T_Type, T_Description
    from tactics
    where common='y';
```

```
BEGIN
for cent in cEntity loop
  eid := cent.E_ID;

  line := 'ENTITY_TYPE '||TO_CHAR(cent.E_ID)||' '|| cent.E_Name;
  insert into images values (lino, line);
  lino := lino + 1;

  line := '  PUBLIC '||cent.E_Public;
  insert into images values (lino, line);
  lino := lino + 1;

  line := '  IDENTITY '||cent.E_Identity;
  insert into images values (lino, line);
  lino := lino + 1;

  for cap in cCapability loop
    cid := cap.C_ID;

    line := '  CAPABILITY '||cap.C_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    line := '    CAPABILITY_DESCRIPTION '||cap.C_Description;
    insert into images values (lino, line);
    lino := lino + 1;

    for dt_cap in cDT_cap loop
      line := '    DATA TABLE '||dt_cap.DT_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := '  END CAPABILITY '||cap.C_Type;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  for sus in cSusc loop
    cid := sus.S_ID;

    line := '  SUSCEPTIBILITY '||sus.S_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    line := '    SUSCEPTIBILITY_DESCRIPTION '||sus.S_Description;
    insert into images values (lino, line);
    lino := lino + 1;

    for dt_sus in cDT_sus loop
      line := '    DATA TABLE '||dt_sus.DT_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := '  END SUSCEPTIBILITY '||sus.S_Type;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  for app in cAppearance loop
    cid := app.AP_ID;

    line := '  APPEARANCE '||app.AP_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    line := '    APPEARANCE_DESCRIPTION '||app.AP_Description;
    insert into images values (lino, line);
    lino := lino + 1;
    for dt_app in cDT_app loop
      line := '    DATA TABLE '||dt_app.DT_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := '  END APPEARANCE '||app.AP_Type;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  for env in cEnvironment loop
    cid := env.ENV_ID;

    line := '  ENVIRONMENT '||env.ENV_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    line := '    ENVIRONMENT_DESCRIPTION '||env.ENV_Description;
    insert into images values (lino, line);
    lino := lino + 1;

    for dt_env in cDT_env loop
      line := '    DATA TABLE '||dt_env.DT_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := '  END ENVIRONMENT '||env.ENV_Type;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  for tac in cTactic loop
    cid := tac.T_ID;

    line := '  TACTIC '||tac.T_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    line := '    TACTIC_DESCRIPTION '||tac.T_Description;
    insert into images values (lino, line);
    lino := lino + 1;

    for dt_tac in cDT_env loop
      line := '    DATA TABLE '||dt_tac.DT_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := '  END TACTIC '||tac.T_Type;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  for con in cContains loop
    cid := con.C_E_ID;

    line := '  CONTAINS '||con.E_Name;
    insert into images values (lino, line);
    lino := lino + 1;

    for cex in cConex loop
      if cex.CE_Type = '+' then
        line := '    PLUS '||cex.CE_Class||' '||TO_CHAR(cex.CE_ID);
      else
        line := '    MINUS '||cex.CE_Class||' '||TO_CHAR(cex.CE_ID);
      end if;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;
```

```
      line := '  END CONTAINS '||con.E_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

  end loop;

    line := 'COMMON CAPABILITY ';
    insert into images values (lino, line);
    lino := lino + 1;
    for cap in cComCap loop
      cid := cap.C_ID;

      line := '  CAPABILITY '||cap.C_Type;
      insert into images values (lino, line);
      lino := lino + 1;

      line := '    CAPABILITY_DESCRIPTION '||cap.C_Description;
      insert into images values (lino, line);
      lino := lino + 1;

      for dt_cap in cDT_cap loop
        line := '    DATA TABLE '||dt_cap.DT_Name;
        insert into images values (lino, line);
        lino := lino - 1;
      end loop;

      line := '  END CAPABILITY '||cap.C_Type;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := 'END COMMON CAPABILITY ';
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'COMMON SUSCEPTIBILITY ';
    insert into images values (lino, line);
    lino := lino + 1;
    for sus in cComSusc loop
      cid := sus.S_ID;

      line := '  SUSCEPTIBILITY '||sus.S_Type;
      insert into images values (lino, line);
      lino := lino + 1;

      line := '    SUSCEPTIBILITY_DESCRIPTION '||sus.S_Description;
      insert into images values (lino, line);
      lino := lino + 1;

      for dt_sus in cDT_sus loop
        line := '    DATA TABLE '||dt_sus.DT_Name;
        insert into images values (lino, line);
        lino := lino + 1;
      end loop;

      line := '  END SUSCEPTIBILITY '||sus.S_Type;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;
    line := 'END COMMON SUSCEPTIBILITY ';
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'COMMON APPEARANCE ';
    insert into images values (lino, line);
    lino := lino + 1;
    for app in cComApp loop
      cid := app.AP_ID;

      line := '  APPEARANCE '||app.AP_Type;
      insert into images values (lino, line);
    lino := lino + 1;

    line := '    APPEARANCE_DESCRIPTION '||app.AP_Description;
    insert into images values (lino, line);
    lino := lino + 1;

    for dt_app in cDT_app loop
      line := '    DATA TABLE '||dt_app.DT_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := '  END APPEARANCE '||app.AP_Type;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;
  line := 'END COMMON APPEARANCE ';
  insert into images values (lino, line);
  lino := lino + 1;

  line := 'COMMON ENVIRONMENT ';
  insert into images values (lino, line);
  lino := lino + 1;
  for env in cComEnv loop
    cid := env.ENV_ID;

    line := '  ENVIRONMENT '||env.ENV_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    line := '    ENVIRONMENT_DESCRIPTION '||env.ENV_Description;
    insert into images values (lino, line);
    lino := lino + 1;

    for dt_env in cDT_env loop
      line := '    DATA TABLE '||dt_env.DT_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := '  END ENVIRONMENT '||env.ENV_Type;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;
  line := 'END COMMON ENVIRONMENT ';
  insert into images values (lino, line);
  lino := lino + 1;

  line := 'COMMON TACTIC ';
  insert into images values (lino, line);
  lino := lino + 1;
  for tac in cComTac loop
    cid := tac.T_ID;

    line := '  TACTIC '||tac.T_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    line := '    TACTIC_DESCRIPTION '||tac.T_Description;
    insert into images values (lino, line);
    lino := lino + 1;

    for dt_tac in cDT_env loop
      line := '    DATA TABLE '||dt_tac.DT_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := '  END TACTIC '||tac.T_Type;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;
  line := 'END COMMON TACTIC ';
  insert into images values (lino, line);
  lino := lino + 1;

commit;
END;
/

select line from images order by line_no;
select 'END SWDB_BOOK3' from dual;
spool off;


set head off;
set pages 0;
set feedback off;
spool BOOK4
select 'SWDB_BOOK4' from dual;
select 'SWDB_BOOK4_SECURITY_LEVEL '|| sec_class
   from swdb_sec_mods where sec_section =4 and sec_base = 'y';
select sec_class from swdb_sec_mods where sec_section =4 and sec_base = 'n';
delete from images;

DECLARE
  lino number := 0;
  line char(255);
  aid number;
  sid number;
  hid number;
  swid number;
  ifid number;
  dpid number;
  cursor cAssets is
    select A_ID, A_Name, A_Type, A_Identity
    from Assets;
  cursor cSupports is
    select SU_ID, SU_Max_Own, SU_Max_Tracked
    from Supports
    where O_A_ID = aid;
  cursor cConcurrent is
    select SU_EN_Name
    from SU_Conc_ET
    where O_SU_ID = sid;
  cursor cHardware is
    select HW_ID, HW_Name, HW_Type, HW_Ident
    from hardware
    where O_A_ID = aid;
  cursor cHardwareDetail is
    select HWD_Type, HWD_Description
    from HW_Detail
    where O_HW_ID = hid
    order by HWD_Type;
  cursor cSoftware is
    select SW_ID, SW_Name, SW_Type, SW_Build_Date
    from software
    where O_A_ID = aid;
  cursor cSoftwareDetail is
    select SWD_Type, SWD_Description
    from SW_Detail
    where O_SW_ID = swid
    order by SWD_Type;
  cursor cIncludeFile is
    select IF_ID, IF_Type, IF_Content, IF_File_Type
    from include_files
    where O_A_ID = aid;
  cursor cFileDetail is
    select DF_Name, DF_Size
    from IF_Detail
    where O_IF_ID = ifid;
  cursor cFileSupport is
    select IFS_Type, IFS_Description
    from IF_Support
    where O_IF_ID = ifid
    order by IFS_Type;
  cursor cPersonnel is
    select  P_Type, P_Description, P_Qualifications
    from Personnel
    where O_A_ID = aid;
  cursor cDisProtocol is
    select DP_ID, DP_Type, DP_Version, DP_OOP, DP_ERROR, DP_PDU_Imp
    from DIS_Protocols
    where O_A_ID = aid;
  cursor cDISPDU is
    select DPT_Name, DPT_SR, DPT_Rate, DPT_Kind
    from DIS_PDU_TYPES
    where O_DP_ID = dpid
    and DPT_Kind = 'dis';
  cursor cISOProtocol is
    select IP_Layer, IP_Protocol, IP_Layer_Name, IP_PDU_Name, IP_Service
    from ISO_Protocols
    where O_A_ID = aid
    order by IP_Layer;

BEGIN
for cA in cAssets loop
  aid := cA.A_ID;

  line := 'ASSET_TYPE '||TO_CHAR(aid)||' '||cA.A_Name;
  insert into images values (lino, line);
  lino := lino + 1;

  line := 'IDENTITY '||cA.A_Identity;
  insert into images values (lino, line);
  lino := lino + 1;

  for cSup in cSupports loop
    sid := cSup.SU_ID;

    line := 'SUPPORTS ENTITY_TYPE';
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'MAX_OWNERSHIP_REPRESENTED '|| TO_CHAR (cSup.SU_Max_Own);
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'MAX_REMOTE_TRACKED '|| TO_CHAR (cSup.SU_Max_Tracked);
    insert into images values (lino, line);
    lino := lino + 1;

    for cCon in cConcurrent loop
      line := 'CONCURRENT_ENTITY_TYPE '|| cCon.SU_EN_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;
  end loop;

  for chw in cHardware loop
    hid := chw.HW_ID;

    line := 'INCLUDES_HARDWARE' || chw.HW_Name;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'HW_Type' || chw.HW_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'HW_ID' || chw.HW_Ident;
    insert into images values (lino, line);
    lino := lino + 1;

    for chd in cHardwareDetail loop
      if chd.HWD_Type = 'desc' then
```

```
        line := 'HW_DESCRIPTION' || chd.HWD_Description;
        insert into images values (lino, line);
        lino := lino + 1;
      end if;
      if chd.HWD_Type = 'peri' then
        line := 'HW_PERIPHERALS' || chd.HWD_Description;
        insert into images values (lino, line);
        lino := lino + 1;
      end if;
      if chd.HWD_Type = 'intf' then
        line := 'HW_INTERFACE' || chd.HWD_Description;
        insert into images values (lino, line);
        lino := lino + 1;
      end if;
    end loop;

    line := 'END_HARDWARE' || chw.HW_Name;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  for csw in cSoftware loop
    swid := csw.SW_ID;

    line := 'INCLUDES_SOFTWARE' || csw.SW_Name;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'SOFTWARE_TYPE' || csw.SW_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'CSCI_BUILD_DATE' || TO_CHAR(csw.SW_Build_Date);
    insert into images values (lino, line);
    lino := lino + 1;

    for csd in cSoftwareDetail loop
      if csd.SWD_Type = 'clp' then
        line := 'CONFIGURATION_LIBRARY_POINTER' || csd.SWD_Description;
        insert into images values (lino, line);
        lino := lino + 1;
      else
        line := 'REQUIRES_DATA_FILE' || csd.SWD_Description;
        insert into images values (lino, line);
        lino := lino + 1;
      end if;
    end loop;

    line := 'END_SOFTWARE' || csw.SW_Name;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  for cif in cIncludeFile loop
    ifid := cif.IF_ID;

    line := 'INCLUDES_DATA_FILE ' || cif.IF_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    for cfd in cFileDetail loop
      line := 'FILE_NAME ' || cfd.DF_Name;
      insert into images values (lino, line);
      lino := lino + 1;

      line := 'FILE_SIZE ' || TO_CHAR(cfd.DF_Size);
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := 'FILE_CONTENT ' || cif.IF_Content;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'FILE_TYPE ' || cif.IF_File_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    for cfs in cFileSupport loop
      line := 'SUPPORTS_'||UPPER(cfs.IFS_Type);
      line := line ||' '||cfs.IFS_Description;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := 'END DATA_FILE';
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  for cp in cPersonnel loop
    line := 'INCLUDES_PERSONNEL '||cp.P_Type;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'PERSONNEL_DESCRIPTION '||cp.P_Description;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'PERSONNEL_QUALIFICATIONS '||cp.P_Qualifications;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'END_PERSONNEL';
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  for cdis in cDisProtocol loop
    dpid := cdis.DP_ID;

    line := 'SUPPORTS_DIS_PROTOCOL '||cdis.DP_Type||' '||cdis.DP_Version;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'OUT_OF_ORDER_PACKETS '|| cdis.DP_OOP;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'ERROR_PACKETS '|| cdis.DP_ERROR;
    insert into images values (lino, line);
    lino := lino + 1;

    for cdp in cDISPDU loop
      line := 'SUPPORTS_DIS_PDU_TYPE ';
      line := line || cdp.DPT_Name || ' ';
      line := line || cdp.DPT_SR || ' ';
      line := line || TO_CHAR(cdp.DPT_Rate);
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := 'END_SUPPORTS_DIS_PROTOCOL';
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  for ciso in cISOProtocol loop
    line := 'SUPPORTS_PROTOCOL '|| ciso.IP_Protocol;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'LAYER '|| TO_CHAR(ciso.IP_Layer)||' ';
    line := line || ciso.IP_Protocol || ' ';
    line := line || ciso.IP_Layer_Name;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'PDU_NAME '|| ciso.IP_PDU_Name;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'SERVICE '|| ciso.IP_Service;
    insert into images values (lino, line);
    lino := lino + 1;

    line := 'END SUPPORTS_PROTOCOL';
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

end loop;
commit;
END;
/

select line from images order by line_no;
select 'END SWDB_BOOK4' from dual;
spool off;


set head off;
set pages 0;
set feedback off;
spool BOOK5
select 'SWDB_BOOK5' from dual;
select 'SWDB_BOOK5_SECURITY_LEVEL '|| sec_class
   from swdb_sec_mods where sec_section =5 and sec_base = 'y';
select sec_class from swdb_sec_mods where sec_section =5 and sec_base = 'n';
delete from images;

DECLARE
  lino number := 0;
  line char(255);
  section number;
  enid number;
  dtid number;
  entid number;
  fields number;
  dims number;

  cursor cSec is
    select sec_class from swdb_sec_mods
    where sec_section = section
    and sec_base = 'n';

  cursor cEnumerated is
    select ET_ID, ET_Name, ET_Item_Size
    from ENUM_TAB;
  cursor cEnuse is
    select EU_User_PDU, EU_User_Record
    from ENUM_USES
    where O_ET_ID = enid;
  cursor cFields is
    select EF_Field_Num, EF_Name, EF_Units, EF_Field_Size
    from ENUM_FIELDS
    where O_ET_ID = enid
    order by EF_Field_Num;
  cursor cValues is
    select EV_Field_Num, EV_Min, EV_Max, EV_Meaning
    from ENUM_VALUES
    where O_ET_ID = enid
    and EV_Min is not null
    order by EV_Field_Num;

  cursor cDataTab is
```

```
    select DT_ID, DT_Name
    from DATA_TAB;
  cursor cDimensions is
    select DD_Dim_Num, DD_Name, DD_Units
    from DATA_DIMS
    where O_DT_ID = dtid
    order by DD_Dim_Num;
  cursor cEntLoop is
    select distinct DE_Entry
    from DATA_Entries
    where O_DT_ID = dtid
    order by DE_Entry;
  cursor cEntries is
    select DE_Dim_Num, DE_Min, DE_Max
    from DATA_ENTRIES
    where O_DT_ID = dtid
    and DE_Entry = entid
    order by DE_Dim_Num;
  cursor cDataVals is
    select DV_Dim_Num, DV_Value
    from DATA_VALUES
    where O_DT_ID = dtid
    and O_DE_Entry = entid
    order by DV_Dim_Num;

BEGIN
  line := 'SWDB_BOOK5_CHAPTER1';
  insert into images values (lino, line);
  lino := lino + 1;

  select sec_class into line from swdb_sec_mods
    where sec_section = 5.1 and sec_base = 'y';
  line := 'SWDB_BOOK5_CHAPTER1_SECURITY_LEVEL '|| line;
  insert into images values (lino, line);
  lino := lino + 1;

  line := '';
  section := 5.1;
  for cS in cSec loop
    line := line || cS.sec_class || ' ';
  end loop;
  insert into images values (lino, line);
  lino := lino + 1;

  for cE in cEnumerated loop
    enid := cE.ET_ID;

    line := 'ENUMERATED_DATA_ITEM '|| cE.ET_Name;
    insert into images values (lino, line);
    lino := lino + 1;

    for cUse in cEnuse loop
      line := 'USED_BY '|| cUse.EU_User_PDU||' '||cUse.EU_User_Record;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := 'DATA_ITEM_SIZE '|| TO_CHAR(cE.ET_Item_Size);
    insert into images values (lino, line);
    lino := lino + 1;

    select count(*) into fields from ENUM_FIELDS where O_ET_ID = enid;
    line := 'NUM_FIELDS '|| TO_CHAR(fields);
    insert into images values (lino, line);
    lino := lino + 1;

    for cF in cFields loop
      line := 'FIELD '||TO_CHAR(cF.EF_Field_Num)||' '||cF.EF_Name;
      line := line ||' '|| cF.EF_Field_Size||' '||cF.EF_Units;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;
    for cV in cValues loop
      line := 'DATA_VALUES';
      insert into images values (lino, line);
      lino := lino + 1;

      line := 'FIELD '||TO_CHAR(cV.EV_Field_Num)||' ';
      line := line || 'MEANING '||TO_CHAR(cV.EV_Field_Num);
      insert into images values (lino, line);
      lino := lino + 1;

      if cV.EV_Max is null then
        line := TO_CHAR(cV.EV_Min)||' '||cV.EV_Meaning;
      else
        line := TO_CHAR(cV.EV_Min)||' - '||TO_CHAR(cV.EV_Max);
        line := line ||' '||cV.EV_Meaning;
        insert into images values (lino, line);
        lino := lino + 1;
      end if;

      line := 'END DATA_VALUES';
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := 'END ENUMERATED_DATA '|| cE.ET_Name;
    insert into images values (lino, line);
    lino := lino + 1;
  end loop;

  line := 'SWDB_BOOK5_CHAPTER2';
  insert into images values (lino, line);
  lino := lino + 1;

  select sec_class into line from swdb_sec_mods
    where sec_section = 5.2 and sec_base = 'y';
  line := 'SWDB_BOOK5_CHAPTER2_SECURITY_LEVEL '|| line;
  insert into images values (lino, line);
  lino := lino + 1;

  line := '';
  section := 5.2;
  for cS in cSec loop
    line := line || cS.sec_class || ' ';
  end loop;
  insert into images values (lino, line);
  lino := lino + 1;

  for cD in cDataTab loop
    dtid := cD.DT_ID;

    line := 'DATA_TABLE '||cD.DT_Name;
    insert into images values (lino, line);
    lino := lino + 1;

    select count(distinct DD_Dim_Num) into dims
      from DATA_DIMS where O_DT_ID = dtid;
    line := 'NUM_DIMENSIONS '||TO_CHAR(dims);
    insert into images values (lino, line);
    lino := lino + 1;

    for cM in cDimensions loop
      line := 'DIMENSION '|| TO_CHAR(cM.DD_Dim_Num);
      line := line||' '||cM.DD_Name;
      if cM.DD_Units is not null then
        line := line ||' ('||cM.DD_Units||')';
      end if;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := 'DATA_TABLE_VALUES ';
```

```
      for cK in cDimensions loop
        line := line || cK.DD_Name || ' ';
      end loop;
      insert into images values (lino, line);
      lino := lino + 1;

      for cE in cEntLoop loop
        entid := cE.DE_Entry;

        for cEL in cEntries loop
          if cEL.DE_Max is null then
            line := TO_CHAR(cEL.DE_Min)||' ';
          else
            line := TO_CHAR(cEL.DE_Min)||'<'||TO_CHAR(cEL.DE_Max)||' ';
          end if;
        end loop;

        for cDV in cDataVals loop
          line := line || TO_CHAR(cDV.DV_Value)||' ';
        end loop;

        insert into images values (lino, line);
        lino := lino + 1;
      end loop;

      line := 'END DATA_TABLE_VALUES';
      insert into images values (lino, line);
      lino := lino + 1;

      line := 'END DATA_TABLE '||cD.DT_Name;
      insert into images values (lino, line);
      lino := lino + 1;
    end loop;

    line := 'END SWDB_BOOK5_CHAPTER2';
    insert into images values (lino, line);
    lino := lino + 1;

    commit;
  END;
  /

  select line from images order by line_no;
  select 'END SWDB_BOOK5' from dual;
  spool off;
```